executes setup display engine **170** in a process block **415**. If the hot key is not pressed during the pre-boot runtime, process **400** proceeds to a process block **460**, wherein the boot target is launched and OS files **150** are loaded into system memory **120**.

In a process block **420**, pre-boot interpreter **165** parses and interprets ACPI namespace **300** to determine current resource settings ("CRS") and possible resource settings ("PRS") of processing system **100**. The CRS and PRS are described by data and control methods encoded in interpreted language code **160** and enumerated in ACPI namespace **300**. The CRS describe current configuration settings of processing system **100** and the PRS describe possible configuration settings of processing system **100**.

Thus, pre-boot interpreter **165** passes the interpreted AML data to setup display engine **170**, which displays the CRS and PRS on a display terminal in a user-friendly format. FIG. **5** illustrates three exemplary user-friendly displays that one embodiment of setup display engine **170** may provide. A display **505** may be the first image the user is shown after pressing the hot key during the pre-boot runtime. By moving a cursor on the screen to highlight "system setup" and pressing "enter", setup display engine **170** generates a display **510**. Display **510** illustrates the CRS. For example, "serial port **1**" is currently assigned to address "0x2F8." By highlighting the "address" with the cursor and pressing "enter", setup display engine **170** generates a display **515**. Display **515** illustrates the PRS for the serial port **1**. Again, the user can effect a configuration change merely by moving the cursor over the desired resource setting and pressing "enter". Upon pressing "enter", setup display engine **170** calls pre-boot interpreter **165** to execute the appropriate AML control methods necessary to effect the configuration change to serial port **140**.

Returning to FIG. **4**, in a process block **425**, setup display engine **170** determines whether a configuration change was requested by the user, such as described above. If a configuration change was requested, process **400** continues to a decision block **430**. If a configuration change was not requested, process **400** proceeds to process block **460**, described above.

In decision block **430**, if the user requested a configuration change to an ACPI compliant hardware device, then process **400** continues to a process block **435** where setup display engine **170** calls pre-boot interpreter **165** to execute the requisite AML control methods to effect the changes in hardware. After effecting the requested change, the boot target is launched in process block **460**.

On the other hand, if the user requested a configuration change to a non-ACPI compliant hardware device in decision block **430**, process **400** proceeds to a process block **445**. In process block **445**, the BIOS executes the requisite legacy APIs to effect the changes in hardware. Once the APIs complete their task, the boot target is launched in process block **460**.

FIG. **6** illustrates one embodiment of a computer system **600** to execute interpreted language code **160** to interact with hardware devices during the pre-boot runtime, in accordance with an embodiment of the present invention. Computer system **600** includes a chassis **605**, a monitor **610**, a mouse **615** (or other pointing device), and a keyboard **620**. The illustrated embodiment of chassis **605** further includes a floppy disk drive **625**, a hard disk **630**, a power supply (not shown), and a motherboard **635** populated with appropriate integrated circuits including system memory **640**, firmware unit **645**, and one or more processors **650**.

In one embodiment, a network interface card ("NIC") (not shown) is coupled to an expansion slot (not shown) of motherboard **635**. The NIC is for connecting computer system **600** to a network **655**, such as a local area network, wide area network, or the Internet. In one embodiment network **655** is further coupled to a remote computer **660**, such that computer system **600** and remote computer **660** can communicate.

Hard disk **630** may comprise a single unit, or multiple units, and may optionally reside outside of computer system **600**. Monitor **610** is included for displaying graphics and text generated by software and firmware programs run by computer system **600**. Mouse **615** (or other pointing device) may be connected to a serial port (e.g., serial port **140** described above), USB port, or other like bus port communicatively coupled to processor(s) **650**. Keyboard **620** is communicatively coupled to motherboard **635** via a keyboard controller (e.g., PS/2 keyboard controller **135** described above) or other manner similar as mouse **615** for user entry of text and commands.

In one embodiment, firmware unit **645** may store interpreted language code **160**, pre-boot interpreter **165**, and setup display engine **170** described above. In one embodiment, hard disk **630** may store OS files **150** and OS interpreter **155** described above. Similarly, system memory **640** may temporarily store ACPI namespace **300** while computer system **600** is in use.

The above description of illustrated embodiments of the invention, including what is described in the Abstract, is not intended to be exhaustive or to limit the invention to the precise forms disclosed. While specific embodiments of, and examples for, the invention are described herein for illustrative purposes, various equivalent modifications are possible within the scope of the invention, as those skilled in the relevant art will recognize.

These modifications can be made to the invention in light of the above detailed description. The terms used in the following claims should not be construed to limit the invention to the specific embodiments disclosed in the specification and the claims. Rather, the scope of the invention is to be determined entirely by the following claims, which are to be construed in accordance with established doctrines of claim interpretation.

What is claimed is:

1. A method, comprising:

providing an interpreted language code to a processing system, the interpreted language code defining how to interact with a hardware device of the processing system; and

interpreting the interpreted language code to interact with the hardware device of the processing system prior to entering an operating system ("OS") runtime mode of operation of the processing system.

2. The method of claim **1** wherein the OS runtime mode of operation is entered no earlier than the beginning of loading an OS into system memory of the processing system.

3. The method of claim **2**, further comprising interpreting the interpreted language code to interact with the hardware device after entering the OS runtime mode of operation.

4. The method of claim **1** wherein the interpreted language code is compliant with an advance configuration and power interface ("ACPI") specification.

5. The method of claim **4** wherein the interpreted language code comprises ACPI machine language("AML").